Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application.

Listing of Claims:

1. (Currently Amended) A method, comprising:

encountering a non-privileged user-level programming instruction;

creating, responsive to the programming instruction, a first shared resource thread (shred) that ~~shares virtual memory address space with~~ belongs to the same process as one or more other shreds; and

executing, responsive to the programming instruction, the first shred concurrently with at least one of the one or more other shreds;

wherein creating the first shred is performed ~~in hardware,~~ without the intervention of an operating system.

2. (Previously Presented) The method of claim 1, further comprising:

maintaining a private state for the first thread, wherein the private state is associated with at least one of a plurality of registers including general purpose registers, floating point registers, MMX registers, segment registers, a flags register, an instruction pointer, control and status registers, SSE registers, and a MXCSR register.

3. (Currently Amended) The method of claim 1, further comprising:

sharing a state among a plurality of shared resource threads ("shreds") associated with a first operating-system-generated thread, the plurality of shreds including the first shred and the one or more other shreds;

while not sharing said state with a second shred, created as a result of a second non-privileged user-level programming instruction, that is associated with a second operating-system-generated thread.

4. (Original) The method of claim 1, further comprising:
sharing a state among a plurality of shreds of the one or more shreds; and
storing the state in one or more registers.

5. (Previously Presented) The method of claim 1, wherein the first shred and the one or more shreds share a current privilege level and share a common address translation.

6. (Previously Presented) The method of claim 1, further comprising:
receiving a non-privileged user-level programming instruction that encodes a shred.

7. (Previously Presented) The method of claim 1, further comprising communicating between the first shred and at least one of the one or more other shreds.

8. (Original) The method of claim 1, further comprising sharing a system state among the one or more shreds.

9. (Previously Presented) The method of claim 7, wherein said communicating is performed via one or more shared registers.

10. (Previously Presented) The method of claim 1, further comprising:
scheduling, responsive to a user-level programming instruction, the first shred and the one or more other shreds without intervention of the operating system.

11. (Previously Presented) The method of claim 7, wherein said communicating is performed via a user-level shred signaling instruction.

12. (Previously Presented) The method of claim 11, further comprising:
storing one or more shred states associated with the one or more shreds responsive to receipt of a context switch request.

13. (Previously Presented) The method of claim 1, further comprising:
handling with user-level exception handler code an exception generated during execution of the first shred, without intervention of the operating system.

14. (Previously Presented) The method of claim 13, further comprising:
receiving the exception from an application program; and
determining whether to report the exception to the operating system.

15. (Previously Presented) The method of claim 1, wherein the shred is to perform input/output (I/O) operations.

16. (Previously Presented) The method of claim 1, wherein the one or more shreds are to perform computation functions.

17. (Currently Amended) An apparatus, comprising:
execution resources to execute a plurality of instructions~~, the execution resources including multiple instruction sequencers~~;
the execution resources to receive a non-privileged user instruction;

the execution resources further to, responsive to the received instruction, ~~begin execution of~~ execute a shared resource thread ("shred") concurrently with one or more other shreds; and

a shared register that is addressable by a user-level instruction, the shared register to provide communication between the shred and the one or more other shreds.

18. (Previously Presented) The apparatus of claim 17, further comprising:
one or more shared shred registers to facilitate communication between two or more of the shreds.

19. Canceled.

20. (Previously Presented) The apparatus of claim 18, wherein the one or more shared registers further comprise a first register that enables an operating system or BIOS to enable multithreading architecture extensions for user-level multithreading.

21. (Previously Presented) The apparatus of claim 17, wherein the execution resources are further to, responsive to the received instruction, begin execution of a shred concurrently with one or more other shreds, without control of an operating system.

22. (Previously Presented) The apparatus of claim 17, wherein the execution resources include one or more processor cores capable of executing multiple shreds concurrently.

23. (Currently Amended) The apparatus of claim 17, further comprising:
[[One]] one or more registers to hold a state shared among the shred and one or more other shreds.

24. (Previously Presented) The apparatus of claim 17, wherein the shred and the one or more other shreds share a current privilege level and share a common address translation.

25. (Previously Presented) The apparatus of claim 17, further comprising logic to execute a user-level instruction to create the shred.

26. (Currently Amended) The apparatus of claim 17, further comprising a mechanism to perform communication between the shred and the one or more other shreds.

27. (Previously Presented) The apparatus of claim 17, further comprising sharing a system state among the shred and the one or more other shreds.

28. (Previously Presented) The apparatus of claim 26, wherein the mechanism further comprises one or more shared registers.

29. Canceled.

30. Canceled.

31. Canceled.

32. (Previously Presented) The apparatus of claim 17, further comprising:
a user-level exception mechanism to report an exception to the shred.

33. (Previously Presented) The apparatus of claim 17, further comprising:
an exception mechanism to report an exception to an operating system.

34. (Previously Presented) The apparatus of claim 32, further comprising:

a mechanism to detect multiple exceptions, each exception associated with a different one of a plurality of concurrently-executing shreds, where the plurality includes the shred and the one or more other shreds;

wherein the exception mechanism includes a prioritizer to prioritize the exceptions;

and wherein the exception mechanism is further to report only one of the prioritized exceptions at a time to the operating system.

35. (Currently Amended) An article of manufacture, comprising:

a machine-accessible medium including data that, when accessed by a machine, cause the machine to perform operations comprising,

receiving user-level programming instructions to execute a plurality of ~~shared resource~~ threads of execution~~(shreds)~~ that each shares a system state with an OS-generated thread;

~~configuring one or more instruction sequencers responsive to the one or more user-level programming instructions;~~

~~scheduling the shreds via hardware;~~ and

executing the plurality of ~~shreds~~ threads of execution concurrently on multiple instruction sequencers.

36. (Previously Presented) The article of manufacture of claim 35, wherein the operations further comprise:

maintaining a private state for each shred of the plurality of shreds, wherein the private state is associated with at least one of a plurality of registers including general purpose registers, floating point registers, MMX registers, segment registers, a flags register, an instruction pointer, control and status registers, SSE registers, and a MXCSR register.

37. (Previously Presented) The article of manufacture of claim 35, wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising sharing a first state among the plurality of shreds, while maintaining a second state privately among an additional_ shred associated with a thread that is not associated with the plurality of shreds, wherein the first state is associated with at least one of a plurality of registers including a control register, a flags register, memory management registers, a local descriptor table register, a task register, debug registers, model specific registers, shared registers, and shred control registers.

38. (Previously Presented) The article of manufacture of claim 35, wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising:

sharing a state among the plurality of shreds; and

storing the state in one or more registers.

39. (Previously Presented) The article of manufacture of claim 35, wherein the plurality of shreds share a current privilege level and share a common address translation.

40. (Previously Presented) The article of manufacture of claim 35, wherein the one or more user-level programming instructions include an instruction to create one or more of the plurality of shreds.

41. (Previously Presented) The article of manufacture of claim 35, wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising communicating among the plurality of shreds.

42. (Previously Presented) The article of manufacture of claim 35, wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising sharing a system state among the plurality of shreds.

43. (Previously Presented) The article of manufacture of claim 35, wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising communicating between the plurality of shreds via one or more shared registers.

44. (Currently Amended) The article of manufacture of claim 35,

wherein an application program controls the plurality of shreds directly, including scheduling of the plurality of shreds, and

wherein an operating system executed by the ~~multiprocessor~~ machine schedules one or more threads.

45. (Previously Presented) The article of manufacture of claim 35, wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising:

associating the plurality of shreds with a thread; and

suspending the plurality of shreds belonging to the thread when a context switch request is received through a single one of the plurality of shreds.

46. (Previously Presented) The article of manufacture of claim 45, wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising:

storing one or more shred states associated with the plurality of shreds when the context switch request is received.

47. (Previously Presented) The article of manufacture of claim 35, wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising:

reporting one or more exceptions to a first shred of the plurality of shreds.

48. (Previously Presented) The article of manufacture of claim 47, wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising:

reporting the one or more exceptions from an application program; and

determining whether to report the one or more exceptions to an operating system.

49. (Currently Amended) The article of manufacture of claim 48, wherein the machine-accessible medium further includes data that causes the machine to perform operations comprising: prioritized-reporting of the one or more exceptions to the operating system; comprising

receiving the one or more exceptions concurrently via different shreds of the plurality of shreds; and

servicing one of the one or more exceptions according to the prioritized-reporting while suspending exception processing of other exceptions of the one or more exceptions.

50. (Previously Presented) The article of manufacture of claim 35, wherein the plurality of shreds perform input/output (I/O) functions and computation functions.

51. (Previously Presented) A system, comprising:

a microprocessor implementing an instruction set architecture (ISA), the microprocessor capable of executing multiple concurrent shreds; and

a memory;

wherein the ISA includes one or more instructions to allow user-level multithreading operations.

52. Canceled.

53. Canceled.

54. Canceled.

55. (Currently amended) A system, comprising:

a microprocessor, including

a plurality of user-level multithreading registers, wherein the registers are addressable by one or more user-level instructions in each of a plurality of user-level threads and are to support communication among the user-level threads ~~coupled to the microprocessor~~; and

memory coupled to the microprocessor, the memory to store the one or more user-level instructions; ~~that stores an instruction set architecture (ISA) compatible with the microprocessor and the plurality of user-level multithreading registers, wherein the memory is from a plurality of memory devices including DRAM, flash, and EEPROM,~~

wherein the microprocessor is further to execute the user-level threads concurrently~~plurality of user-level multithreading registers and the ISA enable multithreading architecture extensions for user-level multithreading~~.

56. (Original) The system of claim 55, wherein the plurality of user-level multithreading registers further comprises a plurality of shared shred registers to facilitate communication between a plurality of shreds and to facilitate synchronization between the plurality of shreds.

57. (Original) The system of claim 56, wherein the plurality of user-level multithreading registers further comprises a plurality of shred control registers to manage the plurality of shreds.

58. (Previously Presented) The system of claim 57, wherein the microprocessor:
receives programming instructions to execute one or more shreds in accordance with the ISA;
configures one or more instruction sequencers via the ISA; and
executes the one or more shreds concurrently.

59. (Previously presented) The apparatus of claim 32, wherein:
the user-level exception mechanism is further to vector to a fixed location in order to allow the shred to service to the exception.

60. (Previously presented) The apparatus of claim 32, wherein:
the plurality of instructions further include a system call instruction to explicitly invoke an operating system to service to the exception.

61. (Currently Amended) The apparatus of claim [[33]]34, wherein said prioritizer employs a round-robin scheme.

62. (Previously presented) The article of manufacture of claim 35, wherein the one or more user-level programming instructions include an instruction to destroy one or more of the plurality of shreds.

63. (Previously presented) The system of claim 51, wherein the one or more instructions include an instruction to create a shred without intervention of an operating system.

64. (Previously presented) The system of claim 51, wherein the one or more instructions include an instruction to destroy a shred without intervention of an operating system.

65. (Previously presented) The system of claim 51, wherein:

the user-level multi-threading operations include concurrent execution of two or more shreds associated with the same thread.

66. (New) The system of Claim 55, wherein the memory is from a plurality of memory devices including DRAM, flash, and EEPROM.

67. (New) A processor, comprising:

execution resources to execute instructions of an instruction set architecture (ISA), the execution resources to execute multiple concurrent shared resource threads ("shreds") that share application state;

wherein the ISA includes one or more non-privileged instructions to allow multithreading operations.

68. (New) The processor of Claim 67, wherein:

the one or more non-privileged instructions include an instruction to create a shared resource thread.